by means of further transformation steps. These transformation steps may start from an automaton as shown in FIG. **10**. As a possible optimization, state transitions according to the schema shown in FIG. **11***a* are identified and are replaced by transitions according to FIG. **11***b*. More in detail, states T of the automaton are identified whose incoming edges, coming, for example, from a state S, are labeled at most with messages of the form (?, A). Furthermore, exactly N (for N>0) transitions (!,B_i) with target state U_i (for $1 \leqq i \leqq N$) are assumed to have their origin in state T. In this situation, state T may be removed by replacing each transition (!, B_i) originating in T and having a target state Us by a transition from S to U_i, said transition being labeled by (?, A)/(!,B_i).

The automata generated by the methods described above can further be simplified in variant embodiments if the state space comprises a control state and a data state and if extended state machines with guarded state transitions and actions for changing the data state are used. Then, the conditions for the sequence descriptions are pairs of a control state and a data state each. Suitable concepts for such extended state machines are known per se, for example from the Statecharts and ROOM formalisms.

In the sample embodiments described above, the sequence descriptions only consisted of communication actions and conditions. In alternative embodiments, however, the method is extended to more powerful formalisms for the description of execution sequences and interactions. For example, sequence descriptions containing choice and repetition operators may be transformed by schematic transformations into the simpler sequence descriptions for which the method has been described above.

According to the sample embodiments described above, when normalizing a component sequence description in sub-steps **26** and **28** (FIG. **4**), missing initial and final conditions have each been replaced by the initial states of the respective components as given in the specification **10** (data **18**). In variant embodiments, each such component sequence description is replaced by a set of sequence descriptions, in each of which an arbitrary condition occurring in the component has been inserted in the place of the missing condition.

It can thus be seen that the invention can be used for creating computer-executable programs in a fully or at least partially automated way. The particulars in the above description of sample embodiments should not be construed as limitations of the scope of the invention, but rather as exemplifications of preferred embodiments thereof. Many other variations are possible and will be readily apparent to persons skilled in the art. Accordingly, the scope of the invention should be determined not by the embodiments illustrated, but by the appended claims and their legal equivalents.

We claim:

1. A method for automatically generating a state-based program for a component of a system consisting of a plurality of components communicating with each other, wherein said program is generated from a specification of said system, said specification comprising interaction-based sequence descriptions of said system, said method comprising the steps of:

a) determining all sequence descriptions of said component defined by said specification of said system,

b) normalizing said sequence descriptions of said component such that a normalized sequence description comprises exactly one initial condition and exactly one final condition and, between said initial condition and said final condition, communication actions only,

c) determining a state-based specification of said component by identifying all equal initial and final conditions of said normalized sequence descriptions of said component with a single state, and

d) determining said state-based program for said component, wherein each sequence description contained in said state-based specification of said component is replaced by a sequence of said communication actions of this sequence description, separated by additionally inserted states.

2. The method of claim **1**, wherein in step a) each sequence description of said system concerning said component is limited to a corresponding sequence description of said component.

3. The method of claim **1**, wherein step b) comprises the following sub-steps:

b1) inserting a condition prescribed in said specification of said system as an initial condition into each of those sequence descriptions of said component that begin with a communication action,

b2) inserting a condition prescribed in said specification of said system as a final condition into each of those sequence descriptions of said component that end with a communication action, and

b3) splitting all sequence descriptions of said component having more than two conditions into a plurality of sequence descriptions of said component, each of the split sequence descriptions having exactly two conditions.

4. The method of claim **3**, wherein at least one of sub-steps b1) and b2) is/are repeated for each of a plurality of conditions to be inserted in order to obtain a corresponding number of sequence descriptions of said component.

5. The method of claim **1**, wherein said interaction-based sequence descriptions of said system are represented by at least one of Message Sequence Charts and UML sequence diagrams.

6. The method of claim **1**, wherein said state-based program for said component is represented by an automaton.

7. The method of claim **6**, wherein said automaton is one of an SDL automaton and a Statechart automaton and a ROOM automaton.

8. The method of claim **1**, wherein said state-based program is based on an extended state automaton whose state space comprises a control state and a data state, and wherein state transitions are performed depending on the current data state and are adapted to change this data state.

9. The method of claim **1**, wherein step d) comprises the step of removing ϵ-transitions from said state-based program.

10. The method of claim **1**, wherein a deterministic state-based program is generated in step d).

11. The method of claim **1**, further including a step of optimizing said state-based program generated in step d).

12. The method of claim **11**, wherein the optimization is performed with respect to the number of states.

13. The method of claim **1**, wherein non-local conditions in said sequence descriptions of said system are replaced in step a) by local conditions in accordance with a predetermined surjective mapping.

14. The method of claim **1**, wherein the method is used for at least one of generating a state-based program for a telecommunication application and generating a state-based program for providing an error tolerant communication of data and generating a state-based program for providing an error tolerant communication of messages and generating a state-based program for a reactive system and generating a